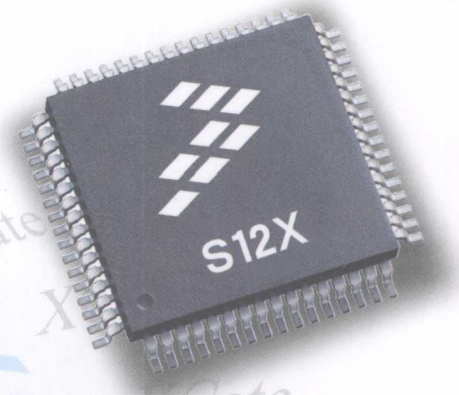
嵌入式系统中的双核技术



邵贝贝　宫　辉　等著

北京航空航天大学出版社

嵌入式系统中的双核技术

邵贝贝　宫　辉　等著

北京航空航天大学出版社

内 容 简 介

本书以16位MC9S12XD/XE系列双核单片机为例，介绍双核单片机的优势和开发方法。从介绍双核单片机的背景知识开始，讲述双核单片机基本硬件系统的设计方法；主处理器初始化协处理器并将部分工作交给协处理器完成的过程；主、协两个处理器的通信机制与克服竞争的方法；用C语言编写双核单片机的应用程序；建立双核单片机应用程序调试的环境；利用商用软件CodeWarrior的教学版本，将嵌入式实时多任务操作系统μC/OS-Ⅱ移植到双核单片机上，建立基于RTOS的开发环境；同时给出了几个实现双核系统应用的范例和源代码，包括利用协处理器提升RTOS性能的方法；还简要介绍了单片机片内容错与纠错技术、双时钟技术和FlexRay通信技术等伴随单片机双核技术发展起来的一些新技术，这些技术的发展与应用将进一步提高嵌入式控制系统的可靠性。

本书可作为相关专业研究生课程教材，也可供理工科大专院校电类本科生和嵌入式控制系统开发应用工程师参考。

图书在版编目(CIP)数据

嵌入式系统中的双核技术/邵贝贝等著. —北京：北京航空航天大学出版社，2008.8

ISBN 978-7-81124-370-3

Ⅰ.嵌… Ⅱ.邵… Ⅲ.微型计算机—系统设计 Ⅳ.TP360.21

中国版本图书馆CIP数据核字(2008)第093149号

嵌入式系统中的双核技术

邵贝贝 宫 辉 等著

责任编辑 王 实

*

北京航空航天大学出版社出版发行

北京市海淀区学院路37号(100083) 发行部电话：010-82317024 传真：010-82328026

http://www.buaapress.com.cn E-mail:bhpress@263.net

涿州市新华印刷有限公司印装 各地书店经销

*

开本：787×960 1/16 印张：21 字数：470千字

2008年8月第1版 2008年8月第1次印刷 印数：5 000册

ISBN 978-7-81124-370-3 定价：35.00元

前　言

随着集成电路技术的高速发展，用于控制领域的单片机内部开始使用双核技术，在主CPU的管理下，另一个CPU内核——协处理器将用来专门处理外部事件。这种双核技术虽然大幅度提升了控制系统的性能，但也增加了应用系统的开发难度。

在嵌入式系统中使用双核技术，是近年来单片机技术发展的一大亮点。本书是介绍双内核单片机开发技术的一本专著，介绍在开发普通单CPU类单片机应用的基础上，如何进一步开发带协处理器的双核单片机。普通单CPU类单片机的开发方法是我们多年来开发嵌入式应用的主要方法，它强调的是在线开发和不使用仿真器。我们在2004年出版的《单片机嵌入式应用的在线开发方法》一书中详尽描述了该方法。本书在此基础上主要增加了协处理器的开发技术，并作为清华大学研究生精品课“嵌入式实时系统与单片机应用”的后续教材之一。

除了讲述双核单片机的开发技术，书中还介绍了开源的实时操作系统(RTOS)μC/OS-Ⅱ的移植和使用。μC/OS-Ⅱ是国内很有影响力的单片机嵌入式RTOS。双核技术的应用，对提高RTOS的实时性、定时精度等技术指标有重要意义。由于最初的μCOS就是为早年传统的MC68HC11单片机写的，与其源码兼容的替代产品——S12系列单片机，是68HC11系列单片机的升级产品，而双核的S12X系列单片机中的一个内核与S12系列单片机兼容。使用或不使用双核单片机中的另外一个内核，μC/OS-Ⅱ都是最佳的RTOS之一。本书还以工程实例的方式，介绍了如何使用协处理器提高μC/OS-Ⅱ的实时性。

这本书也是为嵌入式应用开发工程师写的，由于16位单片机比8位单片机要复杂很多，所以对于那些急于开发产品而又不熟悉16位单片机的工程师，采用我们提供的开发工具，尽快把目标系统开发环境搭建起来，边开发、边学习，是多快好省的办法。本书也可供理工科大专院校电类本科生学习参考。

16位单片机性能强于8位单片机，与32位单片机相比，在运算能力上虽然相对低一些，但在嵌入式应用中却比32位单片机有独到的优势，16位CPU结构简单，寄存器入栈、出栈快，中断响应快。而2个16位CPU比1个32位CPU在嵌入式应用中优势会更加显著。16位单片机使用16位地址总线，指针变量只需要2 B，对内存的占用只有32位单片机的一半，而由此带来的64 KB寻址空间的限制是传统16位单片机的缺憾。以分页管理方式扩展存储空间到8 MB巧妙地解决了这个问题。特别是采用全线性编制方式，借助全局寄存器，使

CPU 可以直接读/写整个存储空间，在保持短指针变量优势的同时，创造性地解决了寻址空间不够的问题，特别适用于量身定制的控制系统的开发。因而作者在此强烈推荐和使用嵌入式应用中 16 位单片机的双核技术。虽然听到一些用户抱怨这种存储管理方式不如 32 位单片机用起来顺手，要求单片机设计人员提供类似 32 位单片机的线性内存编制方式，但这样，16 位单片机的一些优势就丧失了。而目前存在的开发相对复杂的弊端，可以通过软件开发工具的完善予以解决。

本书的第 1 章简单介绍近年来以双核技术为代表的单片机技术发展的一些新亮点，以及双核单片机系列，目的在于强调单片机应用是个性化的，用户针对不同的应用，一定要选择最合适的单片机。第 2 章以带协处理器的 16 位单片机 S12XDP512 和 S12XE100 为例，给出开发双核单片机的一般方法：从设计单片机最小硬件系统开始，实现人与单片机通过串行口对话，尽快让单片机“活”起来。第 3 章和第 4 章分别介绍双核单片机的主处理器和协处理器，以及各自的指令集、汇编语言编程等。第 5 章讲解如何在单片机上建立 C 语言程序的运行环境，C 语言和汇编语言是怎么接口的。第 6 章讲解如何将 μC/OS－Ⅱ移植到 S12X 单片机上。第 7 章介绍后台在线开发(BDM)方法的原理。第 8 章讲解商用软件 CodeWarrior 的使用方法入门。第 9 章通过用协处理器管理异步串行口的实例，给出第 2 个 CPU 内核的开发方法。第 10 章介绍如何使用协处理器提高 μC/OS－Ⅱ的实时性。在附录 A 中，给出相关开发套件及内嵌监控程序的使用方法；附录 B、C 和 D 是便于用户在开发中查阅的指令表。

感谢 Freescale 半导体公司对我校教学与科研的长期支持。感谢何峰、侯磊、卓开阔、冯泽东和谢俊红等研究生在研制 MC9S12XDP512 和 MC9S12XE100 开发板、BDM 调试工具及提供应用范例程序等方面做出的贡献。参与本书编写的还有龚光华、薛涛和曾鸣等。

需求在增长，技术在发展，社会对人才的需求是无止境的，嵌入式系统中的双核技术刚刚起步，作者对其的了解及系统相关知识有限，不当之处请读者指正。关于双核单片机系列及其开发工具和今后的发展，请访问我们的网站：

www. tsinghua－Freescale. com。

作　者

2008 年 5 月于清华大学

目 录

第1章

单片机技术发展新趋势及双核单片机

微控制器(micro-controller)在中国俗称单片机,而单片机(single chip computer)在英语中则很少使用。单片机是在微处理器(micro-processor)的基础上发展起来的。微处理器最早出现在20世纪70年代中期。微处理器需要与存储器和I/O接口电路共同组合成应用系统。这种以微处理器为核心的电子学应用技术称为微机接口技术。

单片机最早出现在20世纪70年代末期,只将部分I/O和少量RAM集成在单片机中,并扩展以紫外线擦除的只读存储器EPROM。程序烧录在单片机外部,方便调试。后来EPROM也集成到单片机中,出现了"窗口片",调试也比较方便。80年代中期后,出现了一种叫做仿真器的单片机调试工具,用微处理器系统或工作在扩展方式下的单片机系统模仿目标单片机的功能,替代目标板上的单片机,以方便调试。在单片机教学中使用仿真器,导致用一种单片机,扩以各种I/O的应用思维方式,这是非常有害的。如果说过去开发掩膜型单片机使用仿真器属不得已,而如今单片机都采用了可以反复擦/写10万次以上的Flash存储器,没有必要再使用仿真器。当前单片机开发技术的趋势是,无论对于8位、16位还是32位机,都会在单片机内部增加1个专门用于调试的模块,该模块以单线通信的方式与外界通信。开发工具仅实现单线通信方式对PC标准接口的转换(USB口或串行口)。通过简单的转换,实现PC对单片机的调试。对于8位、16位和32位单片机,开发工具也在逐渐走向兼容。

如今,在各个行业和领域都能找到单片机的应用实例。人们对现代化、智能化的强烈要求,促使计算机的嵌入式应用迅猛发展。

单片机,顾名思义是将计算机的CPU(微处理器)、存储器(包括随机存储器RAM、只读存储器ROM)和I/O(输入/输出)模块集成在一个电路芯片上,并将应用程序固化在存储器中,再嵌入到产品中去。其应用对象几乎是无限的,故单片机的设计也必须是个性化的。不同的单片机有不同的应用定位,不要企图用一种单片机(例如8051)去适应所有的场合。

实际上单片机有过一个别名,叫做CSIC(用户定义的集成电路)。当某种需求有了一定的数量,用户就可以要求单片机供应商针对这种需求设计一款单片机。这种单片机用某一个成熟的CPU,加上特定数量的存储器RAM和ROM,以及特定的一些I/O模块构成。这种单片机除了用于用户定义的那种有批量的产品,也可以用于其他产品。于是,对于其他用户就又多了一种可选用的单片机。世界上有许多著名的单片机公司,每个公司都有几个、乃至几十个单

片机产品系列，每个单片机系列有几个乃至几十个不同的单片机品种，总可以在现成的产品中找到用户所需要的那一款。CSIC有别于ASIC，ASIC是专用集成电路。所谓专用集成电路是针对专门的产品设计的专用电路，设计与研制的成本较高，并存在一定的风险。因此，选择一款合适的单片机开发的应用系统可以实现低成本且无风险。

近年来，单片机技术的发展出现了一些新趋势，双核技术是其中之一。这些新技术的发展趋势要求应用开发工程师在嵌入式应用系统设计思想上应该有一些新理念，使嵌入式应用系统的可靠性、实时性得到更大提高。应用系统设计工程师无疑需要学习这些新技术，特别是了解这些新技术对应用系统设计者在设计理念方面的影响，以便设计出性能更好的应用系统。这些新技术和新观念主要表现在下面所述的几个方面。

1.1　片上系统以及应用系统单片化趋势

片上系统，也称为SOC(System On Chip)，其优势是显而易见的。它提高了系统的可靠性，降低了应用系统硬件的复杂程度，不但减小了硬件的尺寸，也降低了成本和功耗。

对于控制系统而言，系统的可靠性是第一位的。对一个控制系统的基本要求是在任何情况下都不能“死机”。但是，当存储器不够用时，将存储器扩展到单片机外面，即将系统总线暴露在片外，却在相当大的程度上降低了系统的可靠性，即增加了系统“死机”的可能性。另外，在I/O模块的数目不够用的情况下，通过并口或串口来扩展一两个外加的I/O接口电路模块虽然不一定会影响系统的可靠性，但将存储器扩展到片外，却是以牺牲可靠性为代价的，且不说成本的增加。以存放应用程序的闪存(Flash)为例，图1-1给出了片内Flash的噪声(左图)和片外Flash的噪声之比较。可以看出，片内Flash比片外扩展的Flash抗外界干扰的能力要好得多。

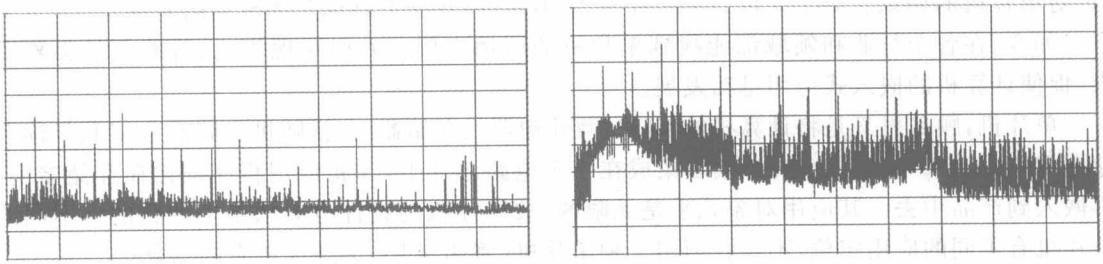


图1-1　片内Flash噪声(左)和片外Flash噪声(右)的比较

随着集成电路技术的发展，集成多个CPU和更多的I/O模块已经不是问题，专门为不同行业、各种要求定制的单片机种类也越来越多，作为嵌入式系统应用设计工程师，可选择的单片机种类也越来越多。在令人眼花缭乱的单片机世界里，从硬件需求来看，不难找到最适用的

单芯片方案。那种以不变应万变，以 1 片 8051 单片机为核心，扩展外围设备接口芯片来适应不同应用的设计理念逐渐被越来越多嵌入式应用系统设计工程师所摒弃，尽量使用单片系统的设计理念得到越来越多的设计工程师认可。

从目前单片机技术发展的特点来看，软件是制约实现单片系统的瓶颈，也就是说，单片机内部可集成的存储器容量还不能很大，特别是片内随机存储器 RAM 的数目还不能做得很大，使得一些对 RAM 需求量大的应用系统，如图像信息的处理等，无法实现应用系统设计的单片化，所幸这类应用系统的特点使其对可靠性的要求不如对控制系统的可靠性要求高。

1.2　以存储器为核心制订解决方案

从统计学的角度看，在嵌入式系统的应用中，程序区 ROM 和数据区 RAM 的比例为4∶1 左右。这个比例来自于早期典型的单板机系统。那时，ROM、RAM 以及 I/O 都是以微处理器为核心外扩的。程序区 ROM 使用掩膜 ROM、可用紫外线擦除的 EPROM 及可用电擦除的 EEPROM；数据区使用静态 RAM，即 SRAM。单片机出现的初期，先是将 100 多字节的 RAM 做到片内，然后是将 ROM 集成到片内的单片机。由于静态 RAM 需要的晶体管数目较多，ROM 和 RAM 的比例开始大于 4∶1。例如，20 世纪 80 年代中期出现的最早的单片机，片内 SRAM 在 128 B 左右，片内 EPROM 在 2 KB 左右，典型的 8 位单片机的 CPU 运算能力在 0.5 MIPS(每秒百万条运算)以下。摩尔定律指出，每 18 个月集成电路的集成度会增长 1 倍。回忆 20 年来单片机发展的历史，单片机片内 CPU 的运算能力大致是按这个规律发展的，甚至还要快。一个单片机内部集成 2 个乃至 4 个 CPU 的多 CPU 技术也已经得到广泛应用。目前，最快的、片内有 4 个 DSP 内核的单片机，运算能力甚至达到了 18 000 MIPS。由于 Flash 技术的发展，片内 ROM 的增加速度也是很快的，大致如摩尔定律预期的。从当初的 1 KB 量级，增加到现在的几 MB 空间，增加了 3 个量级。

在过去 20 年中，单片机片内随机存储器静态 RAM 由于始终没有革命性的突破，与 CPU 运算速度、片内 Flash 容量的增速相比，仅提高了不到 2 个量级。这主要是由于 SRAM 的基本单元是双稳态电路，至少需要 6 只晶体管才能构成 1 个双稳态电路，成本也相对较高，使集成度的提高受到一定的制约。目前，最复杂的 32 位单片机，片内 RAM 最多的是 64 KB；典型的单片机片内 Flash 与 RAM 之比在 16∶1 以上。因此，应用设计工程师只能迁就单片机制造技术的实际情况，尽量减小应用程序对 RAM 数量的需求。

综上所述，20 多年来，CPU 运算能力提高了 4 个量级，ROM 容量增加了 3 个量级，而片内 RAM 仅增加了 2 个量级。片内 RAM 仍然是 SOC 系统的瓶颈。

对存储器的需求，特别是应用程序对 RAM 的需求，可以分为两类：一类是以过程控制为目的的应用，其特点是对系统可靠性要求较高和对系统 RAM 的需求相对有限；另一类是图像处理类应用，至少应该在 RAM 中保存一幅图像。目前，典型的 PC 显示屏幕图像也需要

1 280×1 024 点阵，一般需要用 2 字节表述一个点的属性，如颜色、是否闪烁等，就需要 2.4 MB RAM 来存储。显然，目前的片内 RAM 还做不到。而图像处理类应用对可靠性的要求远小于过程控制类应用，个别像素出错并无大碍。而目前用来扩展 RAM 空间的动态随机存储器 DRAM，1 片就可以做到几 MB。

嵌入式应用系统设计中，越来越多地使用分布式系统，常常将两类应用分开，分别用两类 CPU 完成。对于过程控制类应用，尽量使用单片机，做到 SOC；对于图像处理类应用，用扩展方式扩展几 MB 乃至几十 MB RAM，可以运行 Linux 类的计算机操作系统，甚至可以直接使用嵌入式 PC。对于有低功耗要求的移动式应用，则使用 ARM 类 CPU，扩以 RAM 和 Flash。

这里重点放在可实现 SOC 的控制类应用上，想办法减少 RAM 的用量，做成高可靠性的单片式 SOC 系统。

以上分析表明，单片机设计工程师将会不断使用新技术，增加片内 RAM 的容量；而作为应用工程师，如何将应用系统对存储器容量的要求尽量降到最低，是选择设计方案时的重中之重。

1. 控制 RAM 用量的思路和方法

20 年前，在以 8 位机为主的时代，片内 RAM 只有几百字节。因此，嵌入式应用工程师只能计划使用片内 RAM，开发方式主要使用汇编语言，应用程序的开发很艰苦。后来开始使用 C 语言，特别是近 10 年来，嵌入式实时系统得到广泛应用，各种开发平台比比皆是，RAM、实时操作系统 RTOS(Real Time Operate System)和驱动都具备了，应用程序的开发似乎变简单了，至于 RAM 的用量，也无非是增加一个芯片的成本而已。

笔者始终认为，在可靠性成本和功耗方面，SOC 都比扩展系统要好得多。工程师只要付出更多的智慧，应该能设计出更好的产品。

2. 选择结构简单的 CPU

呼吁在嵌入式系统应用中使用实时操作系统 RTOS，虽然其本身会占用一定的 RAM 空间。RTOS 将一个应用分解成多个任务，每个任务都有自己的栈空间，要占用 RAM 资源。RTOS 做任务切换、任务调度时，要将 CPU 寄存器全部推入当前任务栈，从新任务的堆栈中获取全部 CPU 寄存器的值，并传给 CPU 寄存器。每个任务的栈空间大小，不但要考虑任务切换，而且要考虑中断嵌套层数，当然还有子程序调用嵌套层数和每个任务局部变量的数目等。

单就中断嵌套层数和任务切换而言，RAM 的需求不会少于下面的表达式：

RAM 需求＞保存 CPU 寄存器的需求×中断嵌套层数×任务数

如果选择 32 位 CPU，例如 ARM，任务切换时至少有 17 个 32 位寄存器要入栈、出栈。“保存 CPU 寄存器的需求”这一项为 72 字节。而如果使用 16 位 CPU，例如 Freescale 公司的 S12 CPU，“保存 CPU 寄存器的需求”这一项仅为 9 字节。于是，以上表达式算出的对 RAM 的总需求要相差 8 倍。若单片机片内只有32 KB的 RAM，使用 16 位 CPU 可能 10 KB RAM

即可完成上述功能；若改用 ARM，则需要 80 KB，即使目前 RAM 最大的单片机中也只有 64 KB RAM ，怎么可能做成 SOC 呢？

3. 使用带存储管理的 CPU

在全国大学生智能车模竞赛中，我们选择了有存储器分页管理的 S12 单片机。由于车模的控制比较简单，对分页管理内存的优点还没有使用体会。这种 16 位 CPU，使用 64 KB 的寻址空间，定义一个指针变量时，这个指针变量仅占用 2 字节内存；而使用全程编址的 32 位 CPU，定义一个指针变量时，这个指针变量则要占用 4 字节内存，比 16 位寻址要多占用 1 倍的 RAM 空间，对于子程序调用嵌套也是一样。

分页管理存储空间有一个优点就是，那些等待运行的任务、未进入就绪态的任务和休眠的任务可能会被排除在 64 KB 寻址空间之外，即 CPU 运行当前任务时，可能根本“看不到”那些当前与 CPU 无关的任务。这在系统可靠性方面会有一定的优势。

在带有协处理器的 S12X 系列单片机的 16 位 CPU 中，寻址空间扩展到 8 MB。可以采用 64 KB 寻址空间的 16 位地址，也可通过一个 7 位的全程寄存器对 8 MB 寻址空间统一实现 23 位地址空间寻址。即使全部使用 3 字节指针变量，仍比 32 位指针变量节省 1/4 RAM。

4. 避免使用浮点数和浮点运算的方法

浮点数在科学计算中非常通用，因为它有很宽的数值动态范围和精度范围；然而，在控制领域，则应尽量避免使用浮点数和浮点运算。

首先，浮点数运算是很花时间的，在科学计算中，浮点数运算由专门的协处理器完成。即便如此，主 CPU 仍然需要将要计算的数据传给协处理器，协处理器计算完后再将结果回传给主 CPU，这个数据传输过程也很花时间。而在单片机中，只有很高端的 32 位处理器才有浮点处理器，绝大多数单片机中没有浮点协处理器。

其次，从系统可靠性角度考虑，过宽的数值动态范围对于控制系统并非好事。考虑瞬间强干扰引起数据的出错，若出现在浮点数的阶码上，这个错误就非同小可。如果用传统的平均、滤波等算法处理数据，带来的偏差也会是不可容忍的。虽然可以在程序中以不断判断数据的合理性的方式排除此类错误，但仍不如让数据总是处在合理的动态范围中。

在控制领域，绝大多数数据用 16 位数表示就够用了。如果是无符号数，16 位整数的动态范围是 0～65 535，它表达的精度是 1/65 536，可保证一般运算精度达到万分之一。这在很多场合都够用。如果还不够用，可使用 24 位数或 32 位长整数，运算精度可到亿分之一。

例如，在控制系统中，需要测量一个动态变化的电压值，典型的电压动态范围是 0～15 V。用一个 16 位数表示这个动态范围，相当于 16 位 ADC 的精度，应该说在绝大多数情况下是够用的。可以这样拆分这个 16 位数，高 4 位表示 0～15 V 电压值的整数部分，小数点后面用 12 位分数表示，此时这个 16 位数可表示的电压值范围为

$$0\sim15\ \text{V}+(1/2+1/4+1/8+\cdots+1/4\,096)\ \text{V}=15.999\ 76\ \text{V}$$

这样，即使在多次测量中有某一次出现错误，该次测量的结果也不会超出合理的数值动态范围。所有的运算可以用整数来算，最后将结果右移12位即可。嵌入式系统用单片机往往有分数运算指令，这个16位数可归一化为0～(1－1/65 536)，即0～0.999 984 74。其运算原理和整数运算实际上没有什么区别。同理，仍可使用24位或32位数大幅度提高运算精度。

由于目前单片机中片内Flash存储器已经可以做得很大，可以把有用的数据、曲线事先算好，用查表、插值等方法简化复杂的运算，例如，三角函数表、对数和反对数表等。

上述所有手段，都是为了实现单片机的单片化应用，实现SOC。尽量用1个单片机芯片完成整个应用系统设计，以提高系统的可靠性，降低成本与功耗。

1.3 使用实时操作系统

10余年来，在嵌入式应用系统中使用实时操作系统的理念已经得到越来越多的应用工程师的认可。

过去，由于RTOS的商业软件价格很高，著名的商业RTOS要好几万美元，而其中大部分是不提供源码的，即所谓黑盒子，故不便于学习和使用。另外，早年的单片机存储器资源有限，容不下RTOS本身的开销。而随着半导体技术的飞速发展，片上系统(SOC)成为嵌入式应用的发展趋势，RTOS的使用也越来越普及。

提倡在计算机嵌入式应用中使用实时操作系统，是因为RTOS使CPU利用率最大化。RTOS将应用分解成多个任务，大大简化了应用系统软件的设计；而多任务间可能出现的竞争问题、多任务间通信问题，都有操作系统替用户考虑；RTOS使控制系统的实时性得到保证，可以接近理论上能达到的最高水平；良好的多任务设计，有助于提高系统的稳定性与可靠性，也使应用程序更便于维护与扩展。嵌入式RTOS是多年来计算机专家们潜心研究的成果和智慧的结晶，其应用的范围也不胜枚举。在现代化社会里，可以说，只要能想到的领域都有计算机的嵌入式应用。开发嵌入式应用的工程师应该把计算机专家的研究成果拿来使用，使其开发出来的产品上一个档次。

由于对嵌入式系统的需求越来越复杂，集成电路技术发展又非常快，片上存储器容量不断扩大，使得在片上系统的开发中使用嵌入式RTOS成为可能。各行各业的嵌入式应用系统工程师都是业内的专家，而计算机一般不是他们的本行，只是一种工具。他们不可能将主要精力花在研究操作系统上，而是要把主要精力放在所研究的专业上，他们需要的是把计算机专家们研究的成果直接拿来使用。RTOS可以将一个复杂的应用分解成多个任务，从而简化应用程序的设计，并可以保证系统的实时性达到或接近理论上能达到的最高水平。但多任务本身会带来一系列问题，主要是任务间的竞争、死锁、优先级反转及任务间同步与通信等，这些问题有RTOS为用户考虑了，用户只要知道RTOS的原理，会使用即可。因为RTOS要经过裁剪、固